FOREWORD

This document is a technical Recommendation for use in developing telemetry channel coding systems and has been prepared by the Consultative Committee for Space Data Systems (CCSDS). The telemetry channel coding concept described herein is the baseline concept for spacecraft-to-ground data communication within missions that are cross-supported between Agencies of the CCSDS.

This Recommendation establishes a common framework and provides a common basis for the coding schemes used on spacecraft telemetry streams. It allows implementing organizations within each Agency to proceed coherently with the development of compatible derived Standards for the flight and ground systems that are within their cognizance. Derived Agency Standards may implement only a subset of the optional features allowed by the Recommendation and may incorporate features not addressed by the Recommendation.

Through the process of normal evolution, it is expected that expansion, deletion or modification to this document may occur. This Recommendation is therefore subject to CCSDS document management and change control procedures which are defined in Reference [1].

DOCUMENT CONTROL

Document	Title	Date	. Status/Remarks
—	Recommendation for Space Data System Standards: Telemetry Channel Coding, Issue 1	May 1984	Original Issue
CCSDS 101.0-B-2	Recommendation for Space Data System Standards: Telemetry Channel Coding, Issue 2	January 1987	Current Issue (Supersedes Issue 1)

NOTE: Substantive technical changes from the previous issue are flagged with change bars in the margin.

CONTENTS

REFERENCES

[1] "Procedures Manual for the Consultative Committee for Space Data Systems", Issue 1, Consultative Committee for Space Data Systems, August 1985 or later issue.

[2] *Tracking and Data Relay Satellite System (TDRSS) Users' Guide,* GSFC STDN 101.2, Rev. 5, NASA-Goddard Space Flight Center, Greenbelt, Maryland, September 1984.

[3] "Packet Telemetry", Recommendation CCSDS 102.0-B-2, Issue 2, Blue Book, Consultative Committee for Space Data Systems, January 1987 or later issue.

[4] Perlman, M., and Lee, J., *Reed-Solomon Encoders - Conventional vs. Berlekamp's Architecture,* JPL Publication 82-71, NASA-Jet Propulsion Laboratory, Pasadena, California, December 1, 1982.

[5] "Telemetry Concept and Rationale", CCSDS 100.0-G-1, Issue 1, Green Book, Consultative Committee for Space Data Systems, January 1987 or later issue.

The latest issues of the CCSDS documents may be obtained from the CCSDS Secretariat at the address indicated on page i.

1 INTRODUCTION

1.1 PURPOSE

The purpose of this document is to establish a common Recommendation for space telemetry channel coding systems to provide cross-support among missions and facilities of member Agencies of the Consultative Committee for Space Data Systems (CCSDS.) In addition, it provides focussing for the development of multi-mission support capabilities within the respective Agencies to eliminate the need for arbitrary, unique capabilities for each mission.

Telemetry channel coding is a method by which data can be sent from a source to a destination by processing data so that distinct messages are created which are easily distinguishable from one another. This allows reconstruction of the data with low error probability, thus improving the performance of the channel.

This document was prepared by the CCSDS primarily for the purpose of facilitating the cross-support concept through standardizing key items of data systems compatibility. While the CCSDS has no power of enforcement, it is expected that this recommendation will be incorporated into each respective Agency's data systems standards, and, through them, will apply to all missions that wish to utilize telemetry channel coding for cross-support.

1.2 SCOPE

Several space telemetry channel coding schemes are described in this document. The characteristics of the codes are specified only to the extent necessary to ensure interoperability and cross-support. The specification does not attempt to quantify the relative coding gain or the merits of each approach discussed, nor the design requirements for encoders or decoders. Some performance information is included in Reference [5].

This Recommendation does not require that coding be used on all cross-supported missions. However, for those planning to use coding, the recommended codes to be used are those described in this document.

The rate 1/2 convolutional code recommended for cross-support is described in Section 2, "Convolutional Coding". Depending on performance requirements, this code alone may be satisfactory.

Users of the NASA Tracking and Data Relay Satellite System (TDRSS) may be required to use periodic convolutional interleaving in addition to the convolutional code above. This approach is described in Section 3, "Convolutional Coding with Interleaving for Tracking and Data Relay Satellite Operations".

Where a greater coding gain is needed than can be provided by the convolutional code alone, a standard Reed-Solomon outer code may be concatenated for improved performance. The specification of the Reed-Solomon code selected for cross-support is given in Section 4, "Reed-Solomon Coding". It should be noted that if a spacecraft, utilizing the services of TDRSS, incorporates Reed-Solomon coding, it is the responsibility of the user project to provide the required Reed-Solomon decoding.

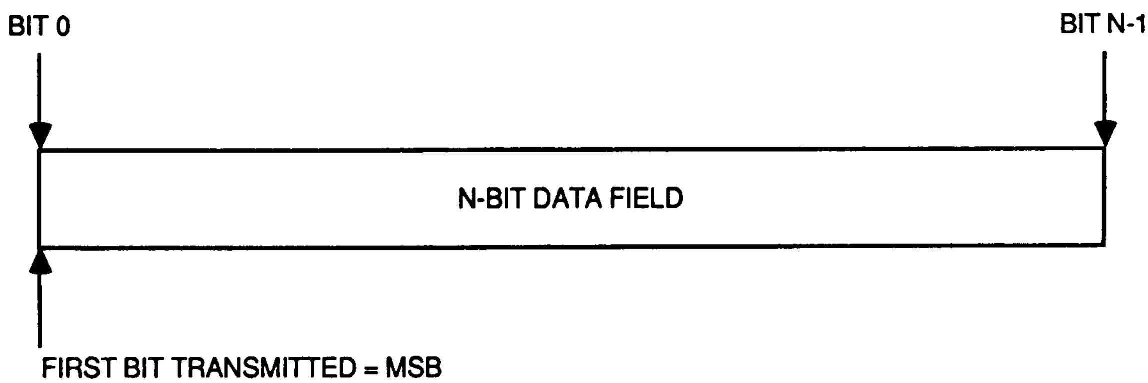
Annex B provides a discussion of the transformation between the Berlekamp and conventional Reed-Solomon symbol representations; Annex C provides a table showing the expansion of the Reed-Solomon coefficients; and Annex D is a glossary of coding terminology used in this document.

1.3 APPLICABILITY

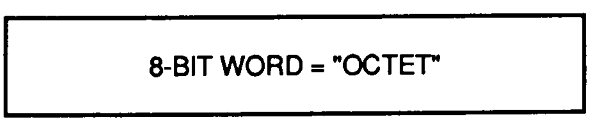
This Recommendation applies to telemetry channel coding applications of space missions anticipating cross-support among CCSDS member Agencies at the coding layer. In addition, it serves as a guideline for the development of compatible internal Agency Standards in this field, based on good engineering practice.

1.4 BIT NUMBERING CONVENTION AND NOMENCLATURE

In this document, the following convention is used to identify each bit in a forward-justified N-bit field. The first bit in the field to be transmitted (i.e., the most left justified when drawing a figure) is defined to be "Bit 0"; the following bit is defined to be "Bit 1" and so on up to "Bit N-1". When the field is used to express a binary value (such as a counter), the Most Significant Bit (MSB) shall be the first transmitted bit of the field, i.e., "Bit 0".



In accordance with modern data communications practice, spacecraft data fields are often grouped into 8-bit "words" which conform to the above convention. Throughout this Recommendation, the following nomenclature is used to describe this grouping:



1.5 RATIONALE

The CCSDS believes it is important to document the rationale underlying the standards chosen, so that future evaluations of proposed changes or improvements will not lose sight of previous decisions. The concept and rationale for Telemetry Channel Coding may be found in Reference [5].

2 CONVOLUTIONAL CODING

The basic code selected for cross-support is a rate 1/2, constraint-length 7 convolutional code. It may be used alone, as described in this section, or in conjunction with enhancements described in the following sections. While slightly different conventions of this code, currently in use by some member Agencies, may continue to be supported for an interim period, it is the recommendation of the CCSDS to universally adopt the single convention described herein.

This recommendation is a non-systematic code and a specific decoding procedure, with the following characteristics:[1]

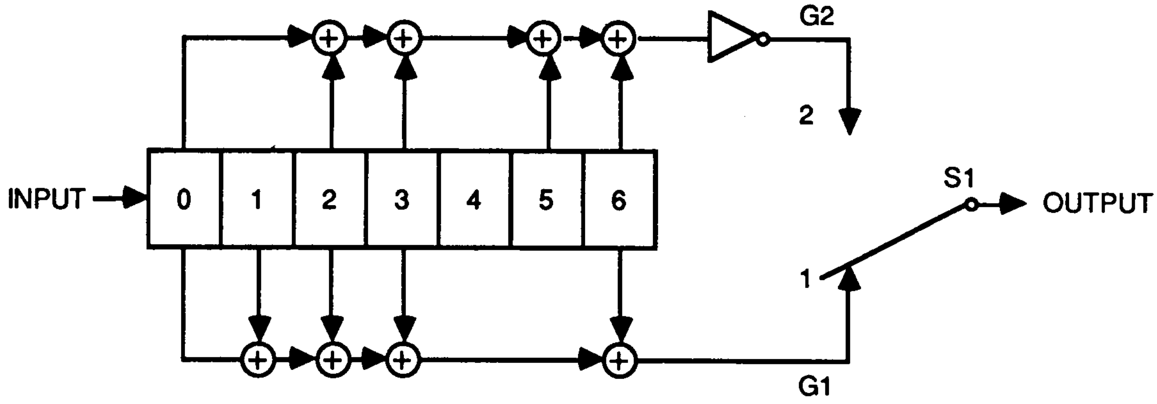
(1)	Nomenclature:	Convolutional code with maximum-likelihood (Viterbi) decoding
(2)	Code rate:	1/2 bit per symbol
(3)	Constraint length:	7 bits
(4)	Connection vectors:	G1 = 1111001; G2 = 1011011
(5)	Phase relationship:	G1 is associated with first Symbol
(6)	Symbol inversion:	On output path of G2

An encoder block diagram with the recommended convention is shown in Figure 2-1.

It is recommended that soft bit decisions with at least 3-bit quantization be used whenever constraints (such as location of decoder) permit.

[1] When suppressed-carrier modulation systems are used, NRZ-M or NRZ-L may be used as a modulating waveform. If the user contemplates conversion of his modulating waveform from NRZ-L to NRZ-M, such conversion should be performed on-board at the input to the convolutional encoder. Correspondingly, the conversion on the ground from NRZ-M to NRZ-L should be performed at the output of the convolutional decoder. This avoids unnecessary link performance loss.

CAUTION: When a fixed pattern in the symbol stream is used to provide node synchronization for the Viterbi decoder, care must be taken to account for any translation of the pattern due to the modulating waveform conversion.



NOTES:

1. = SINGLE BIT DELAY.

2. FOR EVERY INPUT BIT, TWO SYMBOLS ARE GENERATED BY COMPLETION OF A CYCLE FOR S1: POSITION 1, POSITION 2.

3. S1 IS IN THE POSITION SHOWN (1) FOR THE FIRST SYMBOL ASSOCIATED WITH AN INCOMING BIT.

4. $\oplus$ = MODULO-2 ADDER.

5. = INVERTER.

Figure 2-1: Convolutional Encoder Block Diagram

3 CONVOLUTIONAL CODING WITH INTERLEAVING FOR TRACKING AND DATA RELAY SATELLITE OPERATIONS

3.1 INTRODUCTION

Users of the TDRSS S-band Single Access (SSA) Channel, where the channel symbol rate exceeds 300 kilosymbols per second, will be required to employ interleaving in conjunction with the convolutional code which has been described in Section 2. Users are cautioned that if such interleaving is not used under these conditions, the Goddard Space Flight Center Networks Directorate does not guarantee the specified performance and will not be obligated to troubleshoot the system in case of problems (Reference [2]).

It should be noted that this interleaving is totally separate and distinct from the interleaving used in conjunction with the Reed-Solomon code described in Section 4.

3.2 DESCRIPTION

The type of interleaving required is called "Periodic Convolutional Interleaving" and is specified in Appendix J of the TDRSS Users' Guide (Reference [2]).

3.3 BYPASS CAPABILITY

A TDRSS-compatible spacecraft using the Periodic Convolutional Interleaving specified in this section must be capable of bypassing its Periodic Convolutional Interleaver in the event direct support from a non-TDRSS ground tracking station is desired. This is because the interference that this interleaving is designed to protect against is not harmful in this configuration, and, moreover, the necessary de-interleavers do not exist at these ground stations.

4 REED-SOLOMON CODING

4.1 INTRODUCTION

While a convolutional code provides good forward error correction capability in a gaussian noise channel, significant additional improvement (particularly to correct bursts of errors from the Viterbi decoder) can be obtained by concatenating a Reed-Solomon (R-S) code with the convolutional code. The Reed-Solomon code forms the **outer** code, while the convolutional code is the **inner** code. The overhead associated with the Reed-Solomon code is comparatively low, and the improvement in the error performance can often provide the nearly error-free channel required to support efficient automated ground handling of space mission telemetry.

The user is cautioned that the R-S outer code described in this section is not intended for use except when concatenated with the inner convolutional code described in Section 2.

4.2 SPECIFICATION

The parameters of the selected Reed-Solomon code are as follows:

(1) J = 8 bits per R-S symbol.

(2) E = 16 R-S symbol error correction capability within a Reed-Solomon codeword.

(3) General characteristics of Reed-Solomon codes:

(a) J, E, and I (the depth of interleaving) are independent parameters.

(b) $n = 2^J - 1 = 255$ symbols per R-S codeword.

(c) 2E is the number of R-S symbols among n symbols of an R-S codeword representing checks.

(d) k = n-2E is the number of R-S symbols among n R-S symbols of an R-S codeword representing information.

(4) Field generator polynomial:

$$F(x) = x^8 + x^7 + x^2 + x + 1$$

over GF(2).

(5) Code generator polynomial:

$$g(x) = \prod_{j=112}^{143} (x - \alpha^{11j}) = \sum_{i=0}^{32} G_i x^i$$

over $GF(2^8)$, where $F(\alpha) = 0$.

It should be recognized that α^{11} is a primitive element in $GF(2^8)$ and that $F(x)$ and $g(x)$ characterize a (255,223) Reed-Solomon code.

(6) The selected code is a systematic code. This results in a systematic codeblock.

(7) Symbol Interleaving

The recommended value of interleaving depth is I=5, but I=1 is permitted.[2]

Symbol interleaving is accomplished in a manner functionally described with the aid of Figure 4-1. (It should be noted that this functional description does not necessarily correspond to the physical implementation of an encoder.)

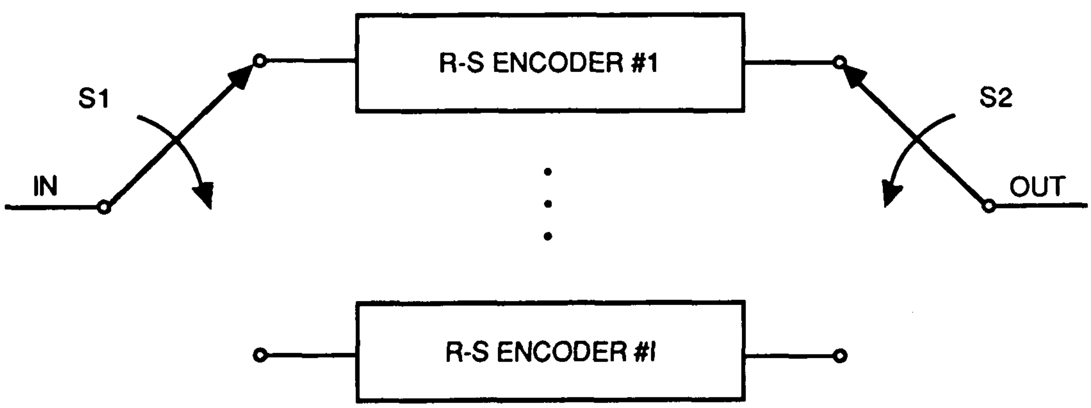


Figure 4-1: Functional Representation of R-S Interleaving

Data bits to be encoded into a single Reed-Solomon Codeblock enter at the port labeled "IN". Switches S1 and S2 are synchronized together and advance from

[2] Users of TDRSS are cautioned that, under some special RFI circumstances, additional measures may have to be employed to obtain a required performance. One such approach may be to utilize a Reed-Solomon code with a different depth of interleaving. In addition to I=1 and I=5, ESA will support I=8.

encoder to encoder in the sequence 1,2, ... I, 1 ... , spending one R-S symbol time (8 bits) in each position.

One codeblock will be formed from 223I R-S symbols entering "IN". In this functional representation, a space of 32I R-S symbols in duration is required between each entering set of 223I R-S information symbols.

Due to the action of S1, each encoder accepts 223 of these symbols, each symbol spaced I symbols apart (in the original stream). These 223 symbols are passed directly to the output of each encoder. The synchronized action of S2 reassembles the symbols at the port labeled "OUT" in the same way as they entered at "IN".

Following this, each encoder outputs its 32 check symbols, one symbol at a time, as it is sampled in sequence by S2.

If, for I=5, the original symbol stream is

$$d_1^1 \ldots d_1^5 \, d_2^1 \ldots d_2^5 \ldots d_{223}^1 \ldots d_{223}^5 \qquad \text{[32 x 5 spaces]}$$

then the output is the same sequence with the [32 x 5 spaces] filled by the [32 x 5] check symbols as shown below:

$$p_1^1 \ldots p_1^5 \ldots p_{32}^1 \ldots p_{32}^5$$

where

$$d_1^i \, d_2^i \ldots d_{223}^i \;\; p_1^i \ldots p_{32}^i$$

is the R-S codeword produced by the i^{th} encoder. If q virtual fill symbols are used in each codeword, then replace 223 by (223 - q) in the above discussion.

With this method of interleaving, the original kI consecutive information symbols that entered the encoder appear unchanged at the output of the encoder with 2EI R-S check symbols appended.

(8) Maximum Codeblock Length

The maximum codeblock length, in R-S symbols, is given by:

$$L_{max} = nI = (2^J - 1)I = 255I$$

(9) Shortened Codeblock Length[3]

A shortened codeblock length may be used to accommodate frame lengths smaller than the maximum. However, since the Reed-Solomon code is a block code, the decoder must always operate on a full block basis. To achieve a full codeblock, "virtual fill" must be added to make up the difference between the shortened block and the maximum codeblock length. The characteristics and limitations of virtual fill are covered in paragraph (10). Since the virtual fill is not transmitted, both encoder and decoder must be set to insert it with the proper length for the encoding and decoding processes to be carried out properly.

When an encoder (initially cleared at the start of a block) receives kI-Q symbols representing information (where Q, representing fill, is a multiple of I, and is less than kI), 2EI check symbols are computed over kI symbols, of which the leading Q symbols are treated as all-zero symbols. A (nI-Q, kI-Q) shortened codeblock results where the leading Q symbols (all zeros) are neither entered into the encoder nor transmitted.

(10) Partitioning and Virtual Fill

The codeblock is partitioned as shown in Figure 4-2.

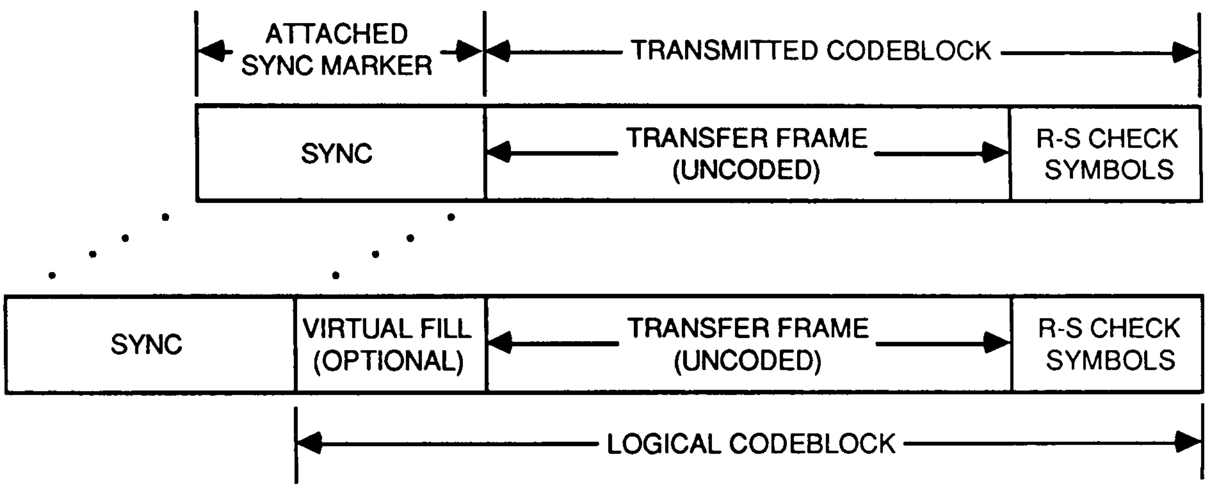


Figure 4-2: Codeblock Partitioning

[3] It should be noted that shortening the transmitted codeblock length in this way changes the overall performance to a degree dependent on the amount of virtual fill used. Since it incorporates no virtual fill, the maximum Packet Telemetry transfer frame length recommended in Reference [3] allows full performance. In addition, as virtual fill in a codeblock is increased (at a specific bit rate), the number of codeblocks per unit time that the decoder must handle increases. Therefore, care should be taken so that the maximum operating speed of the decoder (codeblocks per unit time) is not exceeded.

The **Reed-Solomon Check Symbols** consist of the trailing 2EI symbols (2EIJ bits) of the codeblock. For I=5, this is always 1280 bits.

The **Telemetry Transfer Frame** is defined by the CCSDS Recommendation for Packet Telemetry (Reference [3]). Whether used with R-S coding or not, it has a maximum length of 8920 bits, not including the 32-bit Attached Sync Marker.

The **Attached Sync Marker** is a 32-bit pattern specified in paragraph 4.2 (12) as an aid to synchronization, and precedes the Telemetry Transfer Frame (if R-S coding is NOT used) or the Transmitted Codeblock (if R-S coding IS used). Frame synchronizers should, therefore, be set to expect a marker at every Telemetry Transfer Frame + 32 bits (if not R-S coded,) or at every Transmitted Codeblock + 32 bits.

The **Transmitted Codeblock** consists of the Telemetry Transfer Frame (without the 32-bit sync marker) and R-S check symbols. It is the received data entity physically fed into the R-S decoder. For I=5 and if no virtual fill is used, the length of the transmitted codeblock will be 10,200 bits; if virtual fill is used, it may be incrementally shorter, depending on the amount used.

The **Logical Codeblock** is the logical data entity operated upon by the R-S decoder. It can have a different length than the transmitted codeblock because it accounts for the amount of virtual fill that was introduced. For I=5, the logical codeblock always appears to have exactly 10,200 bits in length.

Virtual fill is used to logically complete the codeblock and is not transmitted. If used, virtual fill shall:

(a) Consist of **all zeros.**

(b) **Not be transmitted.**

(c) Not change in length during a tracking pass.

(d) Be inserted only at the **beginning** of the codeblock (i.e., after the attached sync marker but before the beginning of the transmitted codeblock).

(e) Be inserted only in **integer multiples** of 8I bits.

(11) Symbol representation and ordering for transmission

Each 8-bit Reed-Solomon symbol is an element of the finite field GF(256). Since GF(256) is a vector space of dimension 8 over the binary field GF(2), the actual 8-bit representation of a symbol is a function of the particular basis that is chosen.

One basis for GF(256) over GF(2) is the set $\{1, \alpha^1, \alpha^2, ..., \alpha^7\}$. This means that any element of GF(256) has a representation of the form

$$u_7\alpha^7 + u_6\alpha^6 + ... + u_1\alpha^1 + u_0\alpha^0$$

where each u_i is either a zero or a one.

There is another basis $\{l_0, l_1, ..., l_7\}$ called the "dual basis" to the above basis. It has the property that

$$\mathrm{Tr}\ (l_i\, \beta^j) = \begin{cases} 1, \text{if } i = j \\ 0, \text{otherwise} \end{cases}$$

where $\beta = \alpha^{117}$, for each j = 0, 1, ... , 7. The function Tr(z), called the "trace", is defined by

$$\mathrm{Tr}\ (z) = \sum_{k=0}^{7} z^{2^k}$$

for each element z of GF(256). Each Reed-Solomon symbol can also be represented as

$$z_0l_0 + z_1l_1 + ... + z_7l_7$$

where each z_i is either a zero or a one. The representation recommended in this document is the **dual basis** eight-bit string $z_0, z_1, ..., z_7$, transmitted in that order (with z_0 first.) The relationship between the two representations is given by the two equations

$$[z_0, \ldots, z_7] = [u_7, \ldots, u_0] \begin{bmatrix} 1&0&0&0&1&1&0&1 \\ 1&1&1&0&1&1&1&1 \\ 1&1&1&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&1&1&1&0&1&0 \\ 1&0&0&1&1&0&0&1 \\ 1&0&1&0&1&1&1&1 \\ 0&1&1&1&1&0&1&1 \end{bmatrix}$$

and

$$[u_7, \ldots, u_0] = [z_0, \ldots, z_7] \begin{bmatrix} 1&1&0&0&0&1&0&1 \\ 0&1&0&0&0&0&1&0 \\ 0&0&1&0&1&1&1&0 \\ 1&1&1&1&1&1&0&1 \\ 1&1&1&1&0&0&0&0 \\ 0&1&1&1&1&0&0&1 \\ 1&0&1&0&1&1&0&0 \\ 1&1&0&0&1&1&0&0 \end{bmatrix}$$

Further information relating the dual basis (Berlekamp) and conventional representations is given in Annex B. Also included is a recommended scheme for permitting the symbols generated in a conventional encoder to be transformed to meet the symbol representation required by this document.

(12) Synchronization

Codeblock synchronization of the Reed-Solomon decoder is achieved by synchronization of the Attached Sync Marker associated with each codeblock.

The recommended 32-bit marker pattern is as follows:

```
0001   1010   1100   1111   1111   1100   0001   1101

FIRST TRANSMITTED              LAST TRANSMITTED
BIT (Bit 0)                    BIT (Bit 31)
```

The pattern may be represented in hexadecimal notation as:

1ACFFC1D

This bit pattern was chosen (Reference [5]) to provide good synchronization properties with a low false alarm probability in a noisy channel under the following conditions: True as well as complemented data sense, forward as well as reverse time ordering, and synchronization directly in the bit domain as well as the symbol domain (as translated by the recommended convolutional code of Section 2).

The marker repeats at every transmitted codeblock length plus 32 bits. For I=5 and no virtual fill, this becomes 10,232 bits.

Since the marker is NOT contained within the encoded data space of the R-S codeblock, it is not presented to the input of the R-S encoder or decoder. This prevents the encoder from routinely regenerating the marker in the check symbol field under certain repeating data-dependent conditions (e.g., a test pattern of 010101010..., among others) which could cause synchronization difficulties at the receiving end.

The relationship between the transfer frame, sync marker, codeblock, and virtual fill is illustrated in Figure 4-2.

(13) Ambiguity Resolution

The ambiguity between true and complemented data must be resolved so that only true data is provided to the Reed-Solomon decoder. Data in NRZ-L form is normally resolved using the 32-bit R-S codeblock marker, while NRZ-M data is self-resolving.

ANNEX B

TRANSFORMATION BETWEEN BERLEKAMP AND CONVENTIONAL REPRESENTATIONS

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

Purpose:

This Annex provides information to assist users of the Reed-Solomon code in this Recommendation to transform between the Berlekamp (dual basis) and Conventional representations. In addition, it shows where transformations are made to allow a conventional encoder to produce the dual basis representation on which the Recommendation is based.

Referring to Figure B-1, it can be seen that information symbols I entering and check symbols C emanating from the Berlekamp R-S encoder are interpreted as

$$[z_0, z_1, \ldots, z_7]$$

where the components z_i are coefficients of l_i, respectively:

$$z_0 l_0 + z_1 l_1 + \ldots + z_7 l_7$$

Information symbols I' entering and check symbols C' emanating from the conventional R-S encoder are interpreted as

$$[u_7, u_6, \ldots, u_0]$$

where the components u_j are coefficients of α^j, respectively:

$$u_7\alpha^7 + u_6\alpha^6 + \ldots + u_0$$

A pre- and post-transformation is required when employing a conventional R-S encoder.

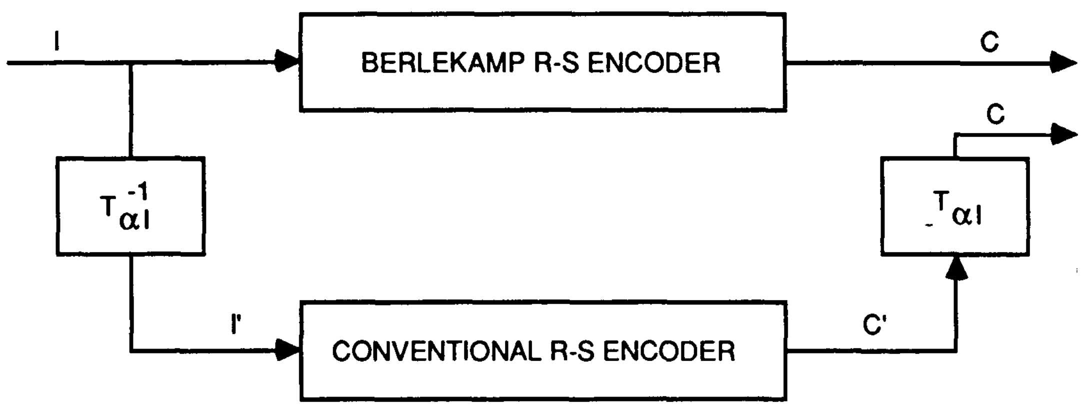


Figure B-1: Transformational Equivalence

Conventional and Berlekamp types of (255,223) Reed-Solomon encoders are assumed to have the same self-reciprocal generator polynomial whose coefficients appear in paragraph 4.2 (4) and (5). The representation of symbols associated with the conventional encoder is the polynomials in "α" appearing in Table B-1, below. Corresponding to each polynomial in "α" is the representation in the dual basis of symbols associated with the Berlekamp type encoder.

Given

$$\alpha^i = u_7\alpha^7 + u_6\alpha^6 + \ldots + u_0$$

where $0 \leq i < 255$ (and α^* denotes the zero polynomial, $u_7, u_6, \ldots = 0, 0, \ldots$), the corresponding element is

$$z = z_0 l_0 + z_1 l_1 + \dots + z_7 l_7$$

where

$$[z_0, z_1, \dots, z_7] = [u_7, u_6, \dots, u_0]\, T_{\alpha l}$$

and

$$T_{\alpha l} = \begin{bmatrix} 1&0&0&0&1&1&0&1 \\ 1&1&1&0&1&1&1&1 \\ 1&1&1&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&1&1&1&0&1&0 \\ 1&0&0&1&1&0&0&1 \\ 1&0&1&0&1&1&1&1 \\ 0&1&1&1&1&0&1&1 \end{bmatrix}$$

Row 1, row 2, ... , and row 8 in $T_{\alpha l}$ are representations in the dual basis of α^7 (10 ... 0), α^6 (010 ... 0), ... , and α^0 (00 ... 01), respectively.

The inverse of $T_{\alpha l}$ is

$$T_{\alpha l}^{-1} = \begin{bmatrix} 1&0&0&0&1&1&0&1 \\ 1&1&1&0&1&1&1&1 \\ 1&1&1&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&1&1&1&0&1&0 \\ 1&0&0&1&1&0&0&1 \\ 1&0&1&0&1&1&1&1 \\ 0&1&1&1&1&0&1&1 \end{bmatrix}$$

Row 1, row 2, ... , and row 8 in $T_{\alpha l}^{-1}$ are polynomials in "α" corresponding to l_0 (10 ... 0), l_1 (010 ... 0), ... , and l_7 (00, ... 01), respectively. Thus,

$$[z_0, z_1, \dots, z_7]\, T_{\alpha l}^{-1} = [u_7, u_6, \dots, u_0]$$

Example 1:

Given information symbol I,

$$[z_0, z_1, \dots, z_7] = 10111001$$

then

$$[1\,0\,1\,1\,1\,0\,0\,1]\ \overset{T_{\alpha l}^{-1}}{\begin{bmatrix} 1&1&0&0&0&1&0&1\\ 0&1&0&0&0&0&1&0\\ 0&0&1&0&1&1&1&0\\ 1&1&1&1&1&1&0&1\\ 1&1&1&1&0&0&0&0\\ 0&1&1&1&1&0&0&1\\ 1&0&1&0&1&1&0&0\\ 1&1&0&0&1&1&0&0 \end{bmatrix}} = [u_7, u_6, \ldots, u_0] = 00101010 = I'$$

Note that the arithmetic operations are reduced modulo 2. Also,

$$[z_0, z_1, \ldots, z_7] = 10111001$$

and

$$[u_7, u_6, \ldots, u_0] = 00101010\ (\alpha^{213})$$

are corresponding entries in Table B-1.

Example 2:

Given check symbol C',

$$[\alpha_7, \alpha_6, \ldots, \alpha_0] = 01011001\ (\alpha^{152})$$

Then,

$$[0\,1\,0\,1\,1\,0\,0\,1]\ \begin{bmatrix} 1&0&0&0&1&1&0&1\\ 1&1&1&0&1&1&1&1\\ 1&1&1&0&1&1&0&0\\ 1&0&0&0&0&1&1&0\\ 1&1&1&1&1&0&1&0\\ 1&0&0&1&1&0&0&1\\ 1&0&1&0&1&1&1&1\\ 0&1&1&1&1&0&1&1 \end{bmatrix} = [z_0, z_1, \ldots, z_7] = 11101000 = C$$

Table B-1: Equivalence of Representations[a]

POWER	POLY IN ALPHA	$\ell_{01234567}$	POWER	POLY IN ALPHA	$\ell_{01234567}$
•	00000000	00000000	31	11001101	01111010
0	00000001	01111011	32	00011101	10011110
1	00000010	10101111	33	00111010	00111111
2	00000100	10011001	34	01110100	00011100
3	00001000	11111010	35	11101000	01110100
4	00010000	10000110	36	01010111	00100100
5	00100000	11101100	37	10101110	10101101
6	01000000	11101111	38	11011011	11001010
7	10000000	10001101	39	00110001	00010001
8	10000111	11000000	40	01100010	10101100
9	10001001	00001100	41	11000100	11111011
10	10010101	11101001	42	00001111	10110111
11	10101101	01111001	43	00011110	01001010
12	11011101	11111100	44	00111100	00001001
13	00111101	01110010	45	01111000	01111111
14	01111010	11010000	46	11110000	00001000
15	11110100	10010001	47	01100111	01001110
16	01101111	10110100	48	11001110	10101110
17	11011110	00101000	49	00011011	10101000
18	00111011	01000100	50	00110110	01011100
19	01110110	10110011	51	01101100	01100000
20	11101100	11101101	52	11011000	00011110
21	01011111	11011110	53	00110111	00100111
22	10111110	00101011	54	01101110	11001111
23	11111011	00100110	55	11011100	10000111
24	01110001	11111110	56	00111111	11011101
25	11100010	00100001	57	01111110	01001001
26	01000011	00111011	58	11111100	01101011
27	10000110	10111011	59	01111111	00110010
28	10001011	10100011	60	11111110	11000100
29	10010001	01110000	61	01111011	10101011
30	10100101	10000011	62	11110110	00111110

[a] From Table 4 of Reference [4].

Note: Coefficients of the "Polynomial in Alpha" column are listed in descending powers of α, starting with α^7.

Table B-1: Cont'd

POWER	POLY IN ALPHA	ℓ 01234567	POWER	POLY IN ALPHA	ℓ 01234567
63	01101011	00101101	95	10111010	10110010
64	11010110	11010010	96	11110011	11011100
65	00101011	11000010	97	01100001	01111000
66	01010110	01011111	98	11000010	11001101
67	10101100	00000010	99	00000011	11010100
68	11011111	01010011	100	00000110	00110110
69	00111001	11101011	101	00001100	01100011
70	01110010	00101010	102	00011000	01111100
71	11100100	00010111	103	00110000	01101010
72	01001111	01011000	104	01100000	00000011
73	10011110	11000111	105	11000000	01100010
74	10111011	11001001	106	00000111	01001101
75	11110001	01110011	107	00001110	11001100
76	01100101	11100001	108	00011100	11100101
77	11001010	00110111	109	00111000	10010000
78	00010011	01010010	110	01110000	10000101
79	00100110	11011010	111	11100000	10001110
80	01001100	10001100	112	01000111	10100010
81	10011000	11110001	113	10001110	01000001
82	10110111	10101010	114	10011011	00100101
83	11101001	00001111	115	10110001	10011100
84	01010101	10001011	116	11100101	01101100
85	10101010	00110100	117	01001101	11110111
86	11010011	00110000	118	10011010	01011110
87	00100001	10010111	119	10110011	00110011
88	01000010	01000000	120	11100001	11110101
89	10000100	00010100	121	01000101	00001101
90	10001111	00111010	122	10001010	11011000
91	10011001	10001010	123	10010011	11011111
92	10110101	00000101	124	10100001	00011010
93	11101101	10010110	125	11000101	10000000
94	01011101	01110001	126	00001101	00011000

Table B-1: Cont'd

POWER	POLY IN ALPHA	ℓ 01234567
127	00011010	11010011
128	00110100	11110011
129	01101000	11111001
130	11010000	11100100
131	00100111	10100001
132	01001110	00100011
133	10011100	01101000
134	10111111	01010000
135	11111001	10001001
136	01110101	01100111
137	11101010	11011011
138	01010011	10111101
139	10100110	01010111
140	11001011	01001100
141	00010001	11111101
142	00100010	01000011
143	01000100	01110110
144	10001000	01110111
145	10010111	01000110
146	10101001	11100000
147	11010101	00000110
148	00101101	11110100
149	01011010	00111100
150	10110100	01111110
151	11101111	00111001
152	01011001	11101000
153	10110010	01001000
154	11100011	01011010
155	01000001	10010100
156	10000010	00100010
157	10000011	01011001
158	10000001	11110110
159	10000101	01101111
160	10001101	10010101
161	10011101	00010011
162	10111101	11111111
163	11111101	00010000
164	01111101	10011101
165	11111010	01011101
166	01110011	01010001
167	11100110	10111000
168	01001011	11000001
169	10010110	00111101
170	10101011	01001111
171	11010001	10011111
172	00100101	00001110
173	01001010	10111010
174	10010100	10010010
175	10101111	11010110
176	11011001	01100101
177	00110101	10001000
178	01101010	01010110
179	11010100	01111101
180	00101111	01011011
181	01011110	10100101
182	10111100	10000100
183	11111111	10111111
184	01111001	00000100
185	11110010	10100111
186	01100011	11010111
187	11000110	01010100
188	00001011	00101110
189	00010110	10110000
190	00101100	10001111

Table B-1: Concluded

POWER	POLY IN ALPHA	$\ell_{01234567}$	POWER	POLY IN ALPHA	$\ell_{01234567}$
191	01011000	10010011	223	01100100	10011010
192	10110000	11100111	224	11001000	10011000
193	11100111	11000011	225	00010111	11001011
194	01001001	01101110	226	00101110	00100000
195	10010010	10100100	227	01011100	00001010
196	10100011	10110101	228	10111000	00011101
197	11000001	00011001	229	11110111	01000101
198	00000101	11100010	230	01101001	10000010
199	00001010	01010101	231	11010010	01001011
200	00010100	00011111	232	00100011	00111000
201	00101000	00010110	233	01000110	11011001
202	01010000	01101001	234	10001100	11101110
203	10100000	01100001	235	10011111	10111100
204	11000111	00101111	236	10111001	01100110
205	00001001	10000001	237	11110101	11101010
206	00010010	00101001	238	01101101	00011011
207	00100100	01110101	239	11011010	10110001
208	01001000	00010101	240	00110011	10111110
209	10010000	00001011	241	01100110	00110101
210	10100111	00101100	242	11001100	00000001
211	11001001	11100011	243	00011111	00110001
212	00010101	01100100	244	00111110	10100110
213	00101010	10111001	245	01111100	11100110
214	01010100	11110000	246	11111000	11110010
215	10101000	10011011	247	01110111	11001000
216	11010111	10101001	248	11101110	01000010
217	00101001	01101101	249	01011011	01000111
218	01010010	11000110	250	10110110	11010001
219	10100100	11111000	251	11101011	10100000
220	11001111	11010101	252	01010001	00010010
221	00011001	00000111	253	10100010	11001110
222	00110010	11000101	254	11000011	10110110

ANNEX C

EXPANSION OF REED-SOLOMON COEFFICIENTS

(THIS ANNEX IS NOT PART OF THE RECOMMENDATION)

Purpose:

While the equations given in the Reed-Solomon Coding Section of this recommendation are fully specifying, this Annex provides additional assistance for those implementing the code.

COEFFICIENTS OF g(x)	POLYNOMIAL IN α:							
	α^7	α^6	α^5	α^4	α^3	α^2	α^1	α^0
$G_0 = G_{32} = \alpha^0$	0	0	0	0	0	0	0	1
$G_1 = G_{31} = \alpha^{249}$	0	1	0	1	1	0	1	1
$G_2 = G_{30} = \alpha^{59}$	0	1	1	1	1	1	1	1
$G_3 = G_{29} = \alpha^{66}$	0	1	0	1	0	1	1	0
$G_4 = G_{28} = \alpha^4$	0	0	0	1	0	0	0	0
$G_5 = G_{27} = \alpha^{43}$	0	0	0	1	1	1	1	0
$G_6 = G_{26} = \alpha^{126}$	0	0	0	0	1	1	0	1
$G_7 = G_{25} = \alpha^{251}$	1	1	1	0	1	0	1	1
$G_8 = G_{24} = \alpha^{97}$	0	1	1	0	0	0	0	1
$G_9 = G_{23} = \alpha^{30}$	1	0	1	0	0	1	0	1
$G_{10} = G_{22} = \alpha^3$	0	0	0	0	1	0	0	0
$G_{11} = G_{21} = \alpha^{213}$	0	0	1	0	1	0	1	0
$G_{12} = G_{20} = \alpha^{50}$	0	0	1	1	0	1	1	0
$G_{13} = G_{19} = \alpha^{66}$	0	1	0	1	0	1	1	0
$G_{14} = G_{18} = \alpha^{170}$	1	0	1	0	1	0	1	1
$G_{15} = G_{17} = \alpha^5$	0	0	1	0	0	0	0	0
$G_{16} = \alpha^{24}$	0	1	1	1	0	0	0	1

Note that $G_3 = G_{29} = G_{13} = G_{19}$.

Further information, including encoder block diagrams, is provided by Perlman and Lee in Reference [4].

ANNEX D

GLOSSARY OF TERMS

(THIS ANNEX **IS NOT** PART OF THE RECOMMENDATION)

Purpose:

This Annex provides definitions for many of the technical terms used in the Recommendation to help clarify their meaning among users and reduce the possibility of misunderstanding among multinational implementers.

GLOSSARY OF TERMS

BINARY SYMMETRIC CHANNEL (BSC):

A channel through which it is possible to send one binary digit per unit of time and for which there is a probability p ($0 < p < 1/2$) that the output is different from the input. This probability does not depend on whether the input is a zero or a one. Successive input digits are affected by the channel independently.

BLOCK ENCODING:

A one-to-one transformation of sequences of length k of elements of a source alphabet to sequences of length n of elements of a code alphabet, $n>k$.

CHANNEL SYMBOL:

The unit of output of the innermost encoder.

CODEBLOCK:

A codeblock of an (n,k) block code is a sequence of n channel symbols which were produced as a unit by encoding a sequence of k information symbols, and will be decoded as a unit.

CODE RATE:

The average ratio of the number of binary digits at the input of an encoder to the number binary digits at its output.

CODEWORD:

In a block code, one of the sequences in the range of the one-to-one transformation (see **Block Encoding**).

CONCATENATION:

The use of two or more codes to process data sequentially with the output of one encoder used as the input of the next.

CONNECTION VECTOR:

In convolutional coding, a device used to specify one of the parity checks to be performed on the shift register in the encoder. For a binary constraint length k convolutional code, a connection vector is a k-bit binary number. A "one" in position i (counted from the left) indicates that the output of the i^{th} stage of the shift register is to be used in computing that parity check.

CONSTRAINT LENGTH:

In convolutional coding, the number of consecutive input bits that are needed to determine the value of the output symbols at any time.

CONVOLUTIONAL CODE:

As used in this document, a code in which a number of output symbols are produced for each input information bit. Each output symbol is a linear combination of the current input bit as well as some or all of the previous k-1 bits where k is the constraint length of the code.

GF(n):

The Galois Field consisting of exactly "n" elements.

INNER CODE:

In a concatenated coding system, the last encoding algorithm that is applied to the data stream. The data stream here consists of the codewords generated by the outer decoder.

MODULATING WAVEFORM:

A way of representing data bits ("1" and "0") by a particular waveform.

NRZ-L:

A modulating waveform in which a data "one" is represented by one of two levels, and a data "zero" is represented by the other level.

NRZ-M:

A modulating waveform in which a data "one" is represented by a change in level and a data "zero" is represented by no change in level.

OUTER CODE:

In a concatenated coding system, the first encoding algorithm that is applied to the data stream.

REED-SOLOMON (R-S) SYMBOL:

A set of J bits that represents an element in $GF(2^J)$, the code alphabet of a J-bit Reed-Solomon code.

SYSTEMATIC CODE:

A code in which the input information sequence appears in unaltered form as part of the output codeword.

TRANSPARENT CODE:

A code that has the property that complementing the input of the encoder or decoder results in complementing the output.

VIRTUAL FILL:

In a systematic block code, a codeword can be divided into an information part and a parity (check) part. Suppose that the information part is N symbols long (a symbol is defined here to be an element of the code's alphabet) and that the parity part is M symbols long. A "shortened" code is created by taking only S ($S < N$) information symbols as input, appending a fixed string of length N-S and then encoding in the normal way. This fixed string is called "fill". Since the fill is a predetermined sequence of symbols, it need not be transmitted over the channel. Instead, the decoder appends the same fill sequence before decoding. In this case, the fill is called "Virtual Fill".

END OF DOCUMENT. ❑